REMARKS

Claims 1-2, 4-6, 12-13, 15-18, 21-24, 28-30, 32, 34-40, 42-49, 51-58, and 60-78 were presented for examination, of which claims 1, 5, 12, 17, 24, 32, 34, 43, 52, 61, 66, 70, and 74 are independent. Claims 3, 7-11, 14, 19-20, 25-27, 31, 33, 41, 50 and 59 were previously canceled. Claims 1-2, 4-6, 12-13, 15-18, 21-24, 28-30, 32, 34-40, 42-49, 51-58, and 60-78 stand rejected. Applicants respectfully submit that the pending claims are in condition for allowance, and respectfully request that the Examiner reconsider the outstanding rejections.

I. Claim Rejections under 35 U.S.C. §103

Claims 1-2, 4-6, 12-13, 15-18, 21-24, 28-30, 32, 34-40, 42-49, 51-58, and 60-78 have been rejected under 35 U.S.C. §103(a) as being anticipated by United States Patent Publication No. 2002/0083413 to Kodosky et al. (hereafter "Kodosky") in view of *practical Validation of Model Based Code Generation for Automotive Applications* by Toeppe (hereafter "Toeppe"). Applicants respectfully traverse this rejection.

A. Claims 1-2, 4, and 6

Applicants respectfully submit that Kodosky and Toeppe, alone or in any reasonable combination, do not disclose or suggest that *the function that is represented graphically has a function prototype that specifies a syntax for invoking the function, the function prototype specifying a function name for the function, and the function is defined in the statechart system in a graphical language*, nor *calling the function that is represented graphically by the function name according to the syntax specified by the function prototype from within the graphical representation of the finite state machine*, which are present in independent claim 1.

The present Application is generally directed to graphical representations of functions that may be described, defined, represented, or invoked in a modeling system for finite state machines. As noted in the Application at page 1, "existing statechart systems for modeling finite state machines permit a user to embed textual definitions of functions in a statechart and invoke those functions in the statechart." The present Application allows a user to "represent visually a procedure performed by a function in a statechart system." (Application at page 3, emphasis added). Such a graphically-represented function provides a number of advantages, such as the

ability to leverage the native parser of the statechart environment, rather than an external parser. Further advantages of graphically defined functions for statechart systems are detailed in the Applicants April 21, 2009 Response at pages 15-17.

Kodosky is one example of an environment that utilizes externally-defined textual code (see, e.g., Kodosky at paragraphs [0168]-[0169]). Kodosky is generally directed to a system and method for generating a graphical program in response to state diagram information (Kodosky at Abstract). Kodosky includes a state diagram that specifies a plurality of states and state transitions. A graphical program generator generates a framework for source code that includes placeholders a user fills in with code for states and transitions (Kodosky at Abstract).

1. Neither Kodosky nor Toeppe discloses or suggests that *the function that is represented graphically has a function prototype that specifies a syntax for invoking the function, the function prototype specifying a function name for the function*

Kodosky does not address graphically represented functions like the functions recited in claim 1. For example, the Examiner recognizes that Kodosky does not disclose that *the function that is represented graphically has a function prototype that specifies a syntax for invoking the function, the function prototype specifying a function name for the function* (Office Action at page 4). Instead, the Examiner relies on Toeppe for the above-quoted feature of claim 1.

Toeppe is generally directed to the validation of C code generated from a Computer Aided Control System Design (CACSD) environment (Toeppe at §1, second paragraph, and §4.4). The Examiner cites §§ 4.4 and 6.1, as well as Figures 6-10 of Toeppe as disclosing a graphically defined function with a function prototype. However, none of the cited figures show a graphical function with a function prototype as recited in claim 1. In the Figures, no function name appears to be provided; instead, the elements noted as "a1," "a2," etc. appear to be the names of states in the state diagrams.

Sections 4.4 and 6.1 both mention function prototypes; however, in both instances, Toeppe is referring to functions in C source code, and not a function prototype for *the function that is represented graphically*. At §4.4, Toeppe explicitly says that the function parameters are used for controlling the modularity or partitioning "of the C source" (Toeppe at §4.4, first four

lines). At §6.1, Toeppe mentions a "complete function prototype" in the section entitled "Auxiliary Functionality," which is distinct from the section labeled "Stateflow Blocks" (Toeppe at §6.1).

Indeed, Toeppe does not represent any function graphically, and so Toeppe cannot disclose a function prototype for a function that is represented graphically.

2. Neither Kodosky nor Toeppe discloses or suggests *calling the function that is represented graphically by the function name according to the syntax specified by the function prototype from within the graphical representation of the finite state machine*

Neither Kodosky nor Toeppe discloses or suggests *calling the function that is represented graphically by the function name according to the syntax specified by the function prototype from within the graphical representation of the finite state machine*, as recited in claim 1. The Office Action appears to include a contradiction regarding this feature of claim 1: although the Examiner recognizes that Kodosky does not disclose a function prototype that specifies a syntax or a function name (Office Action at page 4, 4th paragraph), the Examiner also asserts that Kodosky does call a graphically represented function "according to the syntax specified by the function prototype" (Office Action at page 4, 3rd paragraph, emphasis added). Kodosky cannot fail to disclose a function prototype and yet call a function by the prototype at the same time. Indeed, Kodosky does not disclose a function that is represented graphically, including a function prototype and function name, as recited in claim 1. Applicants have maintained this position in previous Responses, and the Examiner has recognized that Kodosky does not disclose a function prototype in several Office Actions (present Office Action at page 4, January 29th, 2009 Office Action at page 4, August 5, 2008 Office Action at page 4). Because Kodosky does not disclose a function prototype, Kodosky cannot disclose calling the graphical function *by the function name according to the syntax specified by the function prototype*.

Toeppe also does not disclose or suggest this feature of claim 1. As noted above, the "functions" referred to in Toeppe are textual functions in the C source. Toeppe's functions are neither *a function that is represented graphically*, nor are they provided *in the graphical representation of the finite state machine*, nor are they called *from within the graphical representation of the finite state machine*, as recited in claim 1. Accordingly, Toeppe does not

disclose or suggest *calling the function that is represented graphically by the function name according to the syntax specified by the function prototype from within the graphical representation of the finite state machine.*

3. Kodosky and Toeppe cannot be combined to result in a graphically represented function that *has a function prototype that specifies a syntax for invoking the function, the function prototype specifying a function name for the function*

Providing textual function prototypes, as described in Toeppe, with the textually-defined functions of Kodosky, would not result in the method of claim 1. Each of Toeppe and Kodoksy utilize textually-defined functions that do not interact with a statechart environment in the same way that a graphically-defined function would. Applicants have previously discussed Kodosky's use of textually-defined functions in detail (see, e.g., Applicants' April 21, 2009 Response at pages 15-17). Toeppe's "functions" are textually-defined functions in C source code (Toeppe at §4.4, first four lines). In contrast, by providing a graphically represented function with a function prototype, as in claim 1, the statechart environment's functionality can be leveraged in ways that are not apparent from Kodosky and Toeppe. For example, the statechart environment can be used to parse, debug, and error-check the function. A combination of the textually-defined functions of Kodosky and Toeppe would not result in a graphically represented function that *has a function prototype that specifies a syntax for invoking the function, the function prototype specifying a function name for the function.*

4. There is no motivation to modify Kodosky in the way suggested by the Examiner

Still further, one having ordinary skill in the art would not modify Kodosky so that *function is defined in the statechart system in a graphical language*, as recited in claim 1. Kodosky is directed to a system that takes a pre-existing state diagram and generates a graphical program generation program (GPG program) from the pre-existing state diagram. Thus, Kodosky is not concerned with the way that functions are defined in the statechart system. Instead, Kodosky is concerned with the product of the statechart system (the state diagram). Accordingly, there is no motivation to modify Kodosky to change the way that functions are defined in a statechart system.

In light of the above, Applicants respectfully submit that Kodosky and Toeppe, alone or in any reasonable combination, do not disclose or suggest the features of claim 1. Claims 2, 4, and 6 depend from claim 1, and therefore include each and every element of claim 1. Applicants respectfully request that the Examiner reconsider and withdraw the 35 U.S.C. §103(a) rejection of claims 1-2, 4, and 6.

B. Claim 5

Independent claim 5 recites that *the function is represented graphically in the statechart system as a diagram comprising graphical elements and has a function prototype that specifies a syntax for invoking the function, the function prototype specifying a function name for the function*. The Examiner recognizes that Kodosky does not disclose or suggest this element of claim 5 (Office Action at page 5). Instead, the Examiner relies on Toeppe for a function prototype. However, as noted above with respect to claim 1, neither Kodosky nor Toeppe discloses graphical functions in a statechart system, nor a function prototype specifying the name for the function that is represented graphically.

Further, as noted above with respect to claim 1, simply combining Toeppe and Kodosky does not result in the graphical functions of claim 1. By providing the graphically represented function with a function prototype, as in claim 5, the statechart environment's functionality can be leveraged in ways not apparent from Kodosky and Toeppe. For example, the statechart environment can be used to parse, debug, and error-check the function. A combination of Kodosky and Toeppe would not result in a graphically represented function that *has a function prototype that specifies a syntax for invoking the function, the function prototype specifying a function name for the function*.

In light of the above, Applicants respectfully submit that Kodosky and Toeppe, alone or in any reasonable combination, do not disclose or suggest the features of claim 5. Applicants respectfully request that the Examiner reconsider and withdraw the 35 U.S.C. §103(a) rejection of claim 5.

C. Claims 12-13, 15-18, 21-23, 24, 28-30, 32, 34-40, 42-49, 51-58, and 60-78

1. Claims 12-13, 15-16, 24, 28-30, 32, 43-49 and 51

Independent claim 12 recites *the graphical function having a function prototype in the statechart system that specifies a syntax for invoking the graphical function, the function prototype specifying a function name for the graphical function.* Independent claims 24 and 32 recite *a function prototype represented in the statechart system.* Independent claim 43 recites *the graphical function having a function prototype in the event-driven system environment.*

As noted above with respect to claims 1 and 5, Toeppe and Kodosky, alone or in any reasonable combination, do not disclose or suggest a function prototype in a statechart or event-driven system environment. Kodosky does not discuss a function prototype at all, and Toeppe's function prototype is in C source code (and not the statechart system or event-driven system)

Claims 13 and 15-16 depend from claim 12, and therefore include each and every element of claim 12. Claims 28-30 depend from claim 24, and therefore include each and every element of claim 24. Claims 44-49 and 51 depend from claim 43, and therefore include each and every element of claim 43.

2. Claims 17-18 and 21-23

Independent claim 17 recites *means to represent the graphical function as an executable state flow diagram in a statechart system.* In light of the above remarks, Applicants respectfully submit that Kodosky and Toeppe, alone or in any reasonable combination, do not represent the graphical function as an executable state flow diagram in a statechart system. Claims 18 and 21-23 depend from claim 17, and therefore include each and every element of claim 17.

3. Claims 34-40 and 42

Independent claim 34 recites that *said function comprises at least two graphical components in the event driven-system modeling environment.* In light of the above remarks, Applicants respectfully submit that Kodosky and Toeppe, alone or in any reasonable combination, do not disclose or suggest a function comprising at least two graphical components in an event-driven system modeling environment. Claims 35-40 and 42 depend from claim 34, and therefore include each and every element of claim 34.

4. Claims 52-58 and 61-65

Independent claim 52 recites *a graphical function defined in the event-driven system environment*. Independent claim 61 recites *graphically representing a function defined in the graphical block diagram environment*.

In light of the above remarks, Applicants respectfully submit that Kodosky and Toeppe, alone or in any reasonable combination, do not disclose or suggest a graphical function defined in the event-driven system environment.

Claims 53-58 and 60 depend from claim 52, and therefore include each and every element of claim 52. Claims 62-65 depend from claim 61, and therefore include each and every element of claim 61.

5. Claims 66-73

Independent claim 66 recites *graphically defining a function in the graphical block diagram environment*. Independent claim 70 recites *a function defined in the graphical block diagram environment*. In light of the above remarks, Applicants respectfully submit that Kodosky and Toeppe, alone or in any reasonable combination, do not disclose or suggest graphically defining a function in a graphical block diagram environment.

Claims 67-69 depend from claim 66, and therefore include each and every element of claim 66. Claims 71-73 depend from claim 70, and therefore include each and every element of claim 70.

6. Claims 74-78

Independent claim 74 recites that *at least a subset of commands of the graphical function are defined through a graphical representation in the graphical block diagram modeling environment*. In light of the above remarks, Applicants respectfully submit that Kodosky and Toeppe, alone or in any reasonable combination, do not disclose or suggest defining at least a subset of commands of the graphical function through a graphical representation in the graphical

block diagram modeling environment. Claims 75-78 depend from claim 74, and therefore include each and every element of claim 74.

In light of the above, Applicants respectfully request that the Examiner reconsider and withdraw the 35 U.S.C. §103(a) rejection of claims 12-13, 15-18, 21-23, 24, 28-30, 32, 34-40, 42-49, 51-58, and 60-78.

CONCLUSION

In view of the above, Applicants believe the pending application is in condition for allowance and urge the Examiner to pass the claims to allowance. Should the Examiner feel that a teleconference would expedite the prosecution of this Application, the Examiner is urged to contact the Applicants' attorney at (617) 227-7400.

Please charge any shortage or credit any overpayment of fees to our Deposit Account No. 12-0080 under Order No. MWS-070RCE3. In the event that a petition for an extension of time is required to be submitted herewith, and the requisite petition does not accompany this response, the undersigned thereby petitions under 37 C.F.R. §1.136(a) for an extension of time for as many months as are required to render this submission timely. Any fee due is authorized to be charged to the aforementioned Deposit Account.

Dated: August 31, 2009

Respectfully submitted,

Electronic signature: /Kevin J. Canning/
Kevin J. Canning
Registration No.: 35,470
LAHIVE & COCKFIELD, LLP
One Post Office Square
Boston, Massachusetts 02109-2127
(617) 227-7400
(617) 742-4214 (Fax)
Attorney/Agent For Applicant